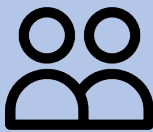


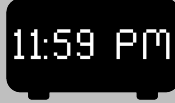
Taller en Sala Nro. 10 Árboles binarios



Los árboles binarios de búsqueda son la base para definir la estructura de datos que utilizan las bases de datos (por ejemplo, MySQL, Oracle, SQL Server) y los sistemas de archivos (por ejemplo, NTFS, EXT4, HFS)



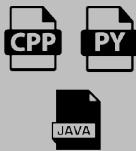
Trabajo en
Parejas



Hoy, plazo
máximo de
entrega



Docente entrega
código suelto en
GitHub



Sí .cpp, .py
o .java



No .zip, .txt,
html o .doc



Alumnos
entregan
código suelto
por GitHub

Ejercicios a resolver

Teniendo en cuenta lo siguiente, por favor respondan los numerales 1,2 y 3:

El siguiente código es la implementación de un árbol binario de búsqueda

```
public class Node {  
  
    public Node left ;  
    public Node right ;  
    public int data ;  
    public Node ( int d ){  
        data = d ;  
    }  
}
```

```
public class BinarySearchTree {  
  
    Node root ;  
    public BinarySearchTree () {  
        root = null ;  
    }  
    public void insertar(int n) {...}  
    public boolean buscar(int n) {...}  
}
```

1. El árbol está triste porque el método buscar no está implementado. El método debe buscar un elemento en el árbol, tendiendo el algoritmo de búsqueda para este tipo de árboles. Implementen el siguiente método:

```
public boolean buscar(int n) {  
  
}
```

2. El árbol sigue triste porque el método insertar no está implementado. El método debe insertar un elemento en el árbol, tendiendo el algoritmo de inserción para este tipo de árboles. Implementen el siguiente método:

```
public void insertar(int n) {  
  
}
```

3. El árbol sigue triste porque el método borrar no está implementado. El método debe borrar un elemento en el árbol. Ideen un algoritmo que preserve la propiedad del árbol de que los mayores van a la derecha y los menores a la izquierda. Implementen el siguiente método:

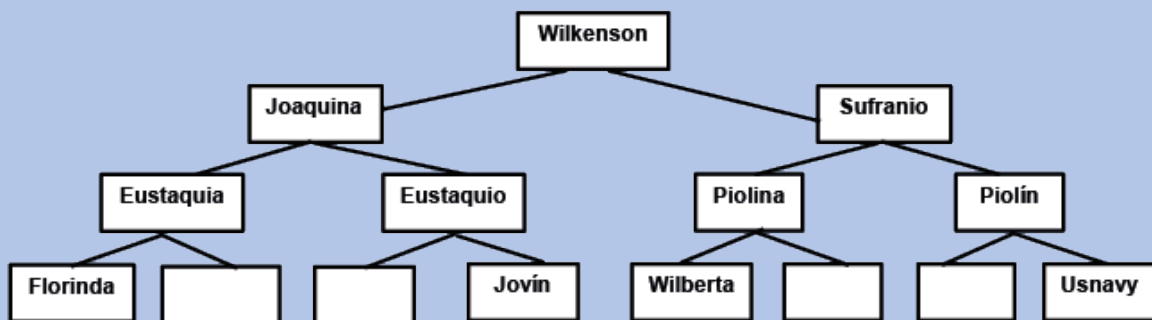
```
public void borrar(int n) {  
  
}
```



Los árboles se utilizan en videojuegos para representar eficientemente el rendering de las escenas y para la detección de colisiones. Para ampliar la información lean en el siguiente enlace: https://en.wikipedia.org/wiki/Binary_space_partitioning

Como un ejemplo, vean aquí: <http://bit.ly/2x05oep>

4. **(Opcional)** Implementen la clase *Nodo* para un árbol binario en el archivo *Nodo.java*
5. **(Opcional)** Implementen la clase *Árbol* para un árbol binario en el archivo *Arbol.java*
6. **(Opcional)** Implementen en Java el siguiente árbol genealógico de ancestros en la clase *Ejemplos*, en el archivo *Ejemplos.java*, dentro del método *main*



7. **(Opcional)** Implementen un método para imprimir en inorden un árbol binario. Este método va en la clase *Arbol*.
8. **(Opcional)** En la clase *Arbol*, implementen un método para dibujar un árbol binario, generando código para esta herramienta: <http://www.webgraphviz.com/>
9. **(Opcional)** Utilicen el método anterior para dibujar el árbol genealógico.

Ayudas para resolver los Ejercicios

| | |
|--|--------------------------------------|
| Ayudas para el Ejercicio 1..... | <u>Pág. 4</u> |
| Ayudas para el Ejercicio 2..... | <u>Pág. 4</u> |
| Ayudas para el Ejercicio 3..... | <u>Pág. 5</u> |

Ayudas para el Ejercicio 1

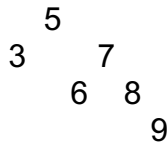


Pista 1: Creen un método auxiliar de la siguiente forma:

```
private boolean buscar(Node nodo, int n)
```



Como un ejemplo, en el árbol siguiente, está el elemento 3, pero no está el elemento 6:



Ayudas para el Ejercicio 2



Pista 1: Creen un método auxiliar de la siguiente forma:

```
private boolean insertar(Node nodo, int n)
```



Pista 2: En una hoja de papel, inserten en un árbol binario ordenado los números del 1 al 10 en este orden:

☒ 1,2,3,4,5,6,7,8,9,10

☒ 5,3,6,2,7,1,9,8,10



Pista 3: Herramientas como BlueJ y Jgrasp les permiten observar gráficamente el árbol

Ayudas para el Ejercicio 3



Pista 1: Creen un método auxiliar de la siguiente forma:

```
private boolean borrar(Node nodo, int n)
```

Ayudas para el Ejercicio 4



Pista 1: Revisen el código *Nodo.java* que están en el GitHub que el docente les entrega

Ayudas para el Ejercicio 5



Pista 1: Revisen el código *Arbol.java* que están en el GitHub que el docente les entrega


Ayudas para el Ejercicio 6



Error Común 1:



DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

| | | |
|---|--|------------------------|
|  | UNIVERSIDAD EAFIT ESCUELA DE INGENIERÍA DEPARTAMENTO DE INFORMÁTICA Y SISTEMAS | Cód. ST0245 |
| | | Estructuras de Datos 1 |

Ayudas para el Ejercicio 7



Pista 1: Este método se usa en el método *main* de la clase *Ejemplos*.

¿Alguna inquietud?

CONTACTO

Docente Mauricio Toro Bermúdez

Teléfono: (+57) (4) 261 95 00 **Ext.** 9473

Correo: mtorobe@eafit.edu.co

Oficina: 19- 627

Agende una cita con él a través de **<http://bit.ly/2gzVg10>** , en la pestaña *Semana*. *Si no da clic en esta pestaña, parecerá que toda la agenda estará ocupada.*